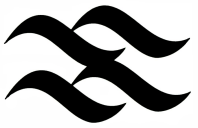
届 别 2014

学 号 201414350352





毕业设计（论文）

基于JAVA的博客论坛网站

姓　　 名 赵洪宇

学 院、 专 业 软件与通信工程学院

导 师 姓 名、职 称 樊洪斌

完 成 时 间

目 录

摘要 ……………………………………………………………………Ⅰ

Abstract ………………………………………………………………Ⅱ

1 概述……………………………………………………………………1

* 1. 国内外研究动态…………………………………………………2
  2. 系统特点和设计基本原则…………………………………… 3

2 系统开发环境及相关技术理论介绍…………………………………4

2.1开发工具intellij idea的介绍……………………………………

2.2 springboot框架介绍………………………………………………

2.3 前端页面显示Vue.js框架介绍……………………………………

2.4 Tomcat容器介绍……………………………………………………

2.5 Mysql数据库介绍…………………………………………………

2.6持久层框架Mybatis介绍……………………………………………

3 系统分析………………………………………………………………5

3.1功能需求分析………………………………………………………6

3.2可行性分析…………………………………………………………7

4 系统总体设计…………………………………………………………8

4.1功能结构的总体框架………………………………………………9

4.2 各功能模块设计…………………………………………………10

5系统详细设计…………………………………………………………11

5.1 系统设计原则及目标……………………………………………12

5.2 系统架构设计及搭建……………………………………………13

5.3数据库设计…………………………………………………………14

6 系统实现……………………………………………………………15

参考文献

摘 要

基于JAVAEE企业级开发的博客网站的研究与开发，我采用的是面向对象的编程思想设计开发出了供学术交流的博客网站，部署在服务器上之后，可以实现兼容各大浏览器（如 谷歌，猎豹，搜狗，火狐），目前的博客网站势力化严重，刷评论，劣质文章，水军，广告商的出现，严重破坏了学术氛围，为此本论文设计了一个供学术交流的博客论坛，从而解决了目前商业化过重的商业网站平台对学术的侵蚀。

论文首先简单介绍了网站的开发环境及相关技术，首先核心技术是JAVA其次是本地化的服务器Tomcat，然后开发平台是Intellij idea2017，最后是数据库部分采用免费开源的Mysql

（关键词）JAVA； Tomcat； Mysql； Intellij idea

**ABSTRACT**

Blog is often known as network log, it is now regarded as a is an expression is a kind of individual thinking and reveal personal style of Internet communication tools, usually set up for personal independence management site. Users are free to post articles they want to publish on their blogs, and the latest articles will be listed at the top of the list. Can't add delete change of post operation, in addition to ordinary visitors provide information display function, blog comments, and comment author and academic interaction function of tourists, in order to improve the blog site customer intimacy. A very mature blog sites including his text edit, upload images, or links to other blog or website, and the associated media, blog has now is an integral part of the social media network. Blog site's explanation is, working software using a computer language, and then to deploy installation on hardware, when users access the specified website will show the user had edited the blog information, easy to build a personal blog on the Internet information of a set of system ". This paper attempts to elaborate the function design, process design, permission design and data model design of personal blog management system from the perspective of software engineering.

**Key words：**JAVA； Tomcat； Mysql； Intellij idea

1 概述

博客,又经常被人们叫做网络的日志，它现在被人们当作是是一种表达个体思想、彰显个人风格的互联网沟通交流工具,通常设置位为个人独立自主管理的网站。使用者可以自由的在自己的博客上发表自己想要发表的文章,最新的文章会被排列在最前面显示,网站上普通的游客只可以浏览博文;不能对博文进行增加删除修改的操作，除了向普通游客用户提供信息展示的功能,博客提供评论、以及回复评论等作者和游客的学术交互功能，以提高博客网站的用户亲密度。一个非常成熟的的博客网站包括了自己编辑的文字、上传的图像、或者其他博客或网站的链接,还有相关联的媒体，博客现在俨然已经是社会媒体网络组成的一部分。博客网站的解释就是,利用计算机语言编写的可运行的软件、然后部署安装到硬件上，当用户访问指定网站的时候,就会显示用户曾经编辑过的博客信息，方便用户在互联网上建立个人博客信息 的一整套系统"。本文尝试从软件工程的角度,详细阐述个人博客管理系统的功能设计、流程设计，权限设计和数据模型设计。

* 1. 国内外研究动态

在当今社会，尤其是互联网高速发展和网民规模数量不断增长的的今天,人们对网络交流工具和互动平台的要求也迅速提高,因此就相应的出现了如论坛、微博、博客、社区、空间等多种多样的的网络互相沟通互动的工具"。博客技术的出现进一步拉近了网络上的人和人们交流的之间的距离,网络上的人们可以轻松、自由、畅所欲言，随时随地的创建、维护开发自己的网络空间,博客这一种社交形式也逐渐成为网民进行在线沟通交流的网络平台。博客,仅仅是一种音译，英文名则为Blogger,他是WebLog的混合到一起的词汇。它的正式名称为网络日志;又被人们音译为部落格或部落阁等,是一种平时的时候由个人开发或者管理、不定期的发布新的文章的网站"。Blog站点所形成的类似于蜘蛛网的网状结构构建成了不同于以前社区的Blog文化,Blog技术成功开创了“博客”这一特殊的文化。个人博客这一技术的出现是现如今网络的热点,越来越多的网民投入到这一个新潮的技术之中，开始把自己内心的想法、感受、心情等用文字、图片或者多媒体的形式发布到网络中，这样更方便通朋友或其他浏览者分享自己的想法感受,让人们极大的体验到了方便快捷的感受。在过去开发的一些网络交流工具和互动平台中常常使用ASP.NET+sqlServer+IIS+JSP，这套技术的缺陷就导致移植性低，开发成本大，前端页面的jsp又是被很多前端人员诟病的技术，因为他将js和html的代码糅合在了一起，使得前端开发人员不便于维护，从而导致代码耦合度高。不便于维护和后期代码的迭代和开发，换到另外一个平台部署又要额外再写一遍代码，尤其是ASP.NET对Linux的支持效果很差。这导致后台的开发web 的主流技术从ASP.NET过渡到了Java，从2000年初的Java过渡到现在，从一开始的原生servlet+原生js代码写网站系统，到后来的springMVC框架+hibernate持久层框架。国内外所有的it爱好者一直关注着技术的进步和需求的进步，然后发展到现在的springBoot框架+mybaits持久层框架+Vue.js前端页面框架，尤其是Vue.js框架的出现，极大的简化了前端页面的开发流程，Vue是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与现代化的工具链以及各种支持类库结合使用时，Vue 也完全能够为复杂的单页应用提供驱动，而为了简化后台代码的开发，由Pivotal团队提供的全新框架springBoot，其设计目的是用来简化应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置,其当初的设计原则就是约定大于配置，所以默认的配置，替代了之前所有的自定义配置，大大简化了架构的配置时间，让开发人员大部分的时间，都能集中在功能的开发和代码的实现上，简短了开发周期和开发成本，博客这一个主题一直都没有变，随着时间的流逝，改变的是技术，技术的逐渐成熟与完善

* 1. 系统特点和设计基本原则

博客网站系统的架构采用了最新最成熟的技术，开发模式上完全采用公司小组开发模式，其特点就是分工明确，敏捷开发，虽然是自己独立开发的网站，但是思想上靠近团队开发的模式，架构上采用springBoot+myBatis+Vue.js，关于网站的页面设计上，在Vue.js的基础上采用了极简的设计模式，尽量给人一种一目了然的感觉，同样也有一种清新的感觉，这种设计思路就是摒弃了现在市场上那些博客平台的广告乱投放的恶现象，业务逻辑设计上，本着学术交流的思想，用户只管理自己的博客文章，可以创建标签，对博客文章进行分类管理，便于其他游客的浏览，博客网站主打沟通交流互动，所以主打评论功能，主张构建一个和谐的学术沟通交流平台

2 系统开发环境及相关技术理论介绍

2.1开发工具intellij idea的介绍

Intellij idea 被许多java开发人员称作是最好，最智能的开发编辑工具，他的出现极大的节省了开发的时间，最大化的提高了开发人员的生产力，毕竟intellij idea 的开发原则就是，减少代码量，让写代码变的更优雅，变成一种艺术，他的智能提示这个功能是减少开发量的一大利器，它提供的智能经验能给予相关建议在每个上下文:即时和智能的代码完成、动态代码分析和可靠的重构工具。关键任务的工具,如集成版本控制系统和各种支持的语言和框架的。它的编辑器甚至涉及到了人体工程学，Intellij idea在每一个设计上尽量减少中断的风险，使得开发人员的流动尽力消除或最小化。IDE会跟随你的背景提出相应的自动工具。他的高支持性和插件化的思想也是被许多开发人员追捧的原因之一，例如SVN(版本控制)，Junit（单元测试），以及许多市面上主流的开发框架，在插件化思想上intellij idea也是做的相当完善，他有一个专门的互动交流社区，供人们分享自己定制的插件，里面功能各异，不同的用户定制化自己的intellij idea ，真正的实现了一千个程序员有一千个版本的Intellij idea，它的高效的整合更是减少了大量的开发时间，一个intellij idea编辑器集成了数据库编辑器，Windows命令行，市面上主流的浏览器，这种跨设备同步IDE的形式，大大减少了程序测试时候的各个窗口轮流切换的时间，减少了不必要的麻烦，从UI设计上来看，intellij idea颠覆了传统编辑器像eclipse那种页面模式，整体设计风格更具有现代科技感。更让大多人耳目一新的感受，如果说只是单独的支持Java这一门语言是不足以火遍全世界的，它同时支持Scala ,CSS,Kolin,Hadoop,Hive等多种语言，这也是全世界为之疯狂的原因之一。

2.2 springboot框架介绍

Springboot是一款优秀的MVC框架，何为MVC，MVC的含义就是相对于应用中的（Model-View-Controller），即把一个应用的输入、处理、输出流程按照Model、View、Controller的方式进行分离，这样一个应用被分成三个层——模型层、视图层、控制层。

视图(View)：代表用户交互界面，MVC设计模式对于视图的处理是限于视图上数据的采集和处理，以及用户的请求，而不包括在视图上的业务流程的处理。

模型(Model)：业务流程/状态的处理以及业务规则的制定。业务流，的处理过程对其它层来说是黑箱操作，模型接受视图请求的数据，并返回最终的处理结果。业务模型的设计可以说是MVC的核心。

控制(Controller)：可以理解为从用户接收请求, 将模型与视图匹配在一起，共同完成用户的请求，但是它却不做任何的数据处理。

说完了MVC就可以聊回springboot了，说springboot是优秀的MVC框架有原因的，在springboot出现之前，长期占据人们视线的是springMVC即便它总被人诟病繁琐的配置文件搞的人晕头转向，但它仍无疑是一款好的数据渲染框架，为此，spring公司开发出了一款完全摒弃XML配置的MVC框架，这样springboot应运而生，它的设计原则是约定大于配置，用户只需要遵守它的默认配置，然后只需要短短的几步就可以搭建一个开发环境，避免了之前繁重的代码和配置文件，而且配置文件还总出错，可以说spring boot的出现大大的减少了数据渲染时带来的不必要的麻烦。最后介绍一下springboot 的执行流程。

1.用户发送请求，例如admin.do

2.发送过来的用户请求通过中心控制器，找到处理器映射器HandlerMapping

3.处理器映射器返回中心控制器一个Handler (处理器)

4.中心控制器找到处理器适配器HandlerAdapter.

5.处理器适配器作用到处理器，处理器开始执行

6.处理器执行之后，返回ModleAndView

7.ModelAndView最终返回到中心控制器。

8.中心控制器找到视图解析器ViewResolver,通过ModelAndView中的view,来找到相应的视图

9.将视图返回到中心控制器。

10.中心控制器会根据返回的ModleAndView中的Model来填充视图解析器返回的View。

11.将渲染后的视图返回给客户端

2.3 前端页面显示Vue.js框架介绍

作为一个渐进式JavaScript框架，vue.js主打的特点就是易用，灵活，高效。Vue.js它支持响应式编程，可以做到保持状态和视图的同步，做到让开发人员实时的看到每次细微改动之后的变化效果，同样组件化的思想也是vue.js设计时的一个考虑的方向，它的思想就是：“一切都是组件”。而且支持组件的多次重复使用，也就是组件可以嵌套其他组件，为了让组件之间能够有效的进行动态的结合在一起，Vue组件可以用props来定义如何接受外部的消息，或者用自定义事件来向外部传递路由消息，这也是vue的特色之一，还有模块化的思想也是深入vue其中，不仅组件之际欸你可以相互套用，每一个单独的独立vue组件都可以看作是一个单独的模块，vue支持用Vue.extend把对象转换为组件构建函数，而且支持导出一个对象，vue和CSS也有很好的相容性，vue文件格式可以支持局部样式，这样做到了每一个vue组件都最终编译成了完全的百分之百的JavaScript模块，而且最重要的是不需要任何浏览器，所以不需要考虑代码的浏览器兼容性，这种高度的自定义方式让你可以在没有给语言块中使用任何你想用的预处理器，同时vue组件支持热替换，酷炫的动画完全可以靠vue实现，而且每次修改代码不用重启服务器，自动加载，vue 的路由方式也是一大特色，它的vue-router这个库可以将以及嵌套进去的路径映射到嵌套的组件，并且为用户提供了详细的路径跳转控制，它的稳定性也是备受青睐的原因，投入使用6年多来达到100%的测试覆盖率，这让Vue在市场上存在着它独特的市场价值

2.4 Tomcat容器介绍

Tomcat是一个servlet容器，初始化端口是8080，它是由sun公司开发完全由java编写实现，tomcat是运行web应用的载体，tomcat是运行在服务器上的小插件，三者的依存关系就是应用依赖tomcat容器，tomcat依赖于服务器，tomcat提供了可以被自己本地部署的服务器的可执行程序代码，而且同时接受来自客户端（用户操作产生）的请求，对使用的开发环境就是tomcat必须支持本地的java开发环境，也就是JDK开发环境，tomcat并不依赖浏览器，他是运行在服务器端的。Tomcat 的基本功能就是接受过来的请求，然后传到tomcat内部，然后调用servlet对象然后解析，这时候容器创建一个servletRequest对象这个对象中包含了之前传递过来的消息。比如，请求头，请求正文，和客户机的IP地址，关于tomcat的文件目录层级关系，Catalina：同开始/关闭shell脚本交互的主类，server：他是整个tomcat组件的容器，包含多个sevice，service：service它包含了connector和container 的集合，service用恰当的connnectot来接受用户端去发起的请求，engine：engine里面包括了host和context，接到发送过来的请求交给相应的host对应的context里面处理，wrapper：wrapper是相对于每个ervlet和container，每个ervlet都有对应的wrapper来管理，同时，tomcat还有一些常用的配置组件，这里只做简单说明，不做深究。

2.5 Mysql数据库介绍

Mysql的特点就是占用服务器内存的体积小，运行起来的时候速度快，总体的来说在开发时候拥有较低的成本，所以被很多公司使用，在全球范围内也占有很大的份额，总的来说，mysql是一个开放开源，时时高效，能承受多线程并发的，并且支持多用户同时登陆的数据库服务器。

mysql是一种关系型数据，官方对关系型数据库的定义就是，把复杂繁复的数据结构，归结为简单的只管的二次元表达关系（也就是人们常说的二维表格形式），在这种关系型数据库中，对想要的数据的操作几乎已经是完全建立在一个或者多个表格上了，让多个表格互相关联，建立某种对应关系，比如两表互相关联，一对多的关系，多对一的关系，或者复杂一点的多对多关系

2.6持久层框架Mybatis介绍

Mybatis是所属于apache公司的一个开源项目，作为一个java的持久层框架，Mybaits提供了持久层框架的关系映射

Mybatis这个框架它自己实现了和方法的接口动态绑定，让它在使用起来更加的方便快捷，它通过xml配置文件，去实现接口中本来要实现的方法，自动生成接口的实现方法，用起来更加的省事儿。

Mybaitis采用的是对象关系映射，将实体类与数据库对应字段进行关系映射，效率更高，对应更准确

Mybaits强大的二级缓存机制能大大提高项目整体的运行效率，和运行速度，让数据保证可靠性的同时，还能高效率。

Mybaits还结合了OGNL(图型导航语言)表达式，这样实现了mybatis的一个最大的特点那就是动态sql，使得编程更灵活，避免了代码与sql语句的耦合。

3 系统分析

3.1 功能需求分析

站在用户角度分析网站的功能需求可以大致的设计出的角色权限吗模式分为普通游客浏览模式，登陆后的管理员模式两种，两种用户模式在具体的功能操作上也有所不同为的就是保护作者的博文的唯一性，只允许作者自己进行修改，而普通游客只有浏览的权限，不过为了实现学术交流讨论的学术氛围，应该为普通游客增添评论功能，这样就可以实现了技术上的交流和思维的碰撞，同时，如果两个人的技术栈大致相同，而且观点也保持一致，还可以为用户们提供添加好友功能，这样就方便了用户们以后的互相交流，既然这个网站主要是针对于管理者设计，毕竟是个人网站博客系统，所以为了方便博主也就是作者的博文管理，需要添加博文的分类管理。根据不同的技术分类成不同的博文，做到为每个博文添加标签，实现分类管理，这样方便于作者的管理，同时也方便于读者的浏览和知识的查找，日志是最重要的部分，他记录了管理者的登陆，退出，等一些列基础的操作信息，这些信息的收集和采集，并且将这些信息展示到网页上有利于管理员对自己的账号的保护，能做到知晓过去一切的操作，防止外来的网络攻击，做到防患于未然。

3.2可行性分析

针对用户里的权限问题，也就是游客和登陆用户的两种身份，可以采取动态展示的设计，根据输入的链接网址不同，然后让用户指向不同的页面，游客只负责浏览，但是登录用户在这之前添加一个登陆页面的验证，验证成功直接跳转到后台管理页面，这一个功能是直接将页面分开写即可，关于后台管理操作部分的大量增删改差操作，可以采取spring boot+mybaits的框架整合到一起，根据前台传送过来的数据，通过springboot 的视图解析器整合成参数，然后传到mybaits，mybatis携带着参数去数据库里面查询，然后根据方法的逻辑执行对应的操作，然后将返回的结果呈现到前台页面，这一工作主要交给vue.js,他负责数据的渲染和数据的展示

4 系统总体设计

4.1 功能结构的总体框架

总体框架采用三层架构模式，即，页面层，业务逻辑层还有针对数据库的数据访问层

详见如图

4.2 各功能模块设计

详见如图

5 系统详细设计

5.1 系统设计原则及目标

低耦合高内聚

5.2 系统架构设计及搭建

项目的整体架构由maven管理，maven是Java项目的一个管理工具，项目中依赖的jar包以及开发环境，都需要在pom.xml中文件标注出来，然后maven会自动导入所有依赖，同时maven还是支持项目的打包操作，将整个复杂的程序打包成一个简单的独立封装的jar包，方便在其他平台或者环境调用，它是整个项目架构的核心，所以在这里，贴出完整的pom文件，即为整个项目所需要的所有架构

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.my.blog.website</groupId>

<artifactId>my-blog</artifactId>

<!-- now -->

<version>3.3.0-SNAPSHOT</version>

<!-- version的时间节点-->

<!-- Tale -->

<!--<version>0.0.1-SNAPSHOT</version>-->

<!-- springboot -->

<!--<version>1.0.1-SNAPSHOT</version>-->

<!-- docker -->

<!--<version>2.0.1-SNAPSHOT</version>-->

<packaging>jar</packaging>

<name>my-blog</name>

<description>my-blog</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.5.1.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<docker.image.prefix>blog</docker.image.prefix>

<docker.plugin.version>0.3.258</docker.plugin.version>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.mybatis.spring.boot</groupId>

<artifactId>mybatis-spring-boot-starter</artifactId>

<version>1.2.0</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter</artifactId>

<exclusions>

<exclusion>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-logging</artifactId>

</exclusion>

</exclusions>

</dependency>

<!-- Start: log4j2 -->

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-slf4j-impl</artifactId>

<version>2.5</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>jcl-over-slf4j</artifactId>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>jul-to-slf4j</artifactId>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>log4j-over-slf4j</artifactId>

</dependency>

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-api</artifactId>

<version>2.5</version>

</dependency>

<dependency>

<groupId>org.apache.logging.log4j</groupId>

<artifactId>log4j-core</artifactId>

<version>2.5</version>

</dependency>

<!-- End: log4j2 -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-websocket</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.commons/commons-lang3 -->

<dependency>

<groupId>org.apache.commons</groupId>

<artifactId>commons-lang3</artifactId>

<version>3.5</version>

</dependency>

<dependency>

<groupId>com.atlassian.commonmark</groupId>

<artifactId>commonmark-ext-gfm-tables</artifactId>

<version>0.8.0</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.mybatis.generator/mybatis-generator-core -->

<!--<dependency>-->

<!--<groupId>org.mybatis.generator</groupId>-->

<!--<artifactId>mybatis-generator-core</artifactId>-->

<!--<version>1.3.5</version>-->

<!--</dependency>-->

<!--过滤emoji字符-->

<dependency>

<groupId>com.vdurmont</groupId>

<artifactId>emoji-java</artifactId>

<version>3.2.0</version>

</dependency>

<!-- https://mvnrepository.com/artifact/com.google.code.gson/gson -->

<dependency>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

<version>2.8.0</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-aop</artifactId>

</dependency>

</dependencies>

<!-- 普通的maven打包方式 -->

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

<!-- 普通的maven打包方式 -->

<!-- 结合docker的maven打包方式,并上传提交创建的镜像 -->

<!--<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

<plugin>

<groupId>com.spotify</groupId>

<artifactId>docker-maven-plugin</artifactId>

<version>${docker.plugin.version}</version>

<executions>

<execution>

<phase>install</phase>

<goals>

<goal>build</goal>

</goals>

</execution>

</executions>

<configuration>

<imageName>${docker.image.prefix}/${project.artifactId}</imageName>

<dockerDirectory>${project.basedir}/docker-extension/my-blog</dockerDirectory>

<resources>

<resource>

<targetPath>/</targetPath>

<directory>${project.build.directory}</directory>

<include>${project.build.finalName}.jar</include>

</resource>

</resources>

</configuration>

</plugin>

</plugins>

</build>-->

<!-- 结合docker的maven打包方式,并上传提交创建的镜像 -->

</project>

5.3数据库设计

数据库名字为tale，监听端口号3306，IP地址为127.0.0.1：3306，tale数据源一共有8张表，其中t\_user表和t\_comments,t\_contents是一对多的关系，即一个用户可以同时对应多篇博文，或者多个评论，t\_metas是文章分类的表，它和t\_comments文章的表对应，互相是多对多的关系，意思为一个文章可以对应多个分类，同时一个分类也可以对应多个标签，也可以理解成双向一对多的关系，然后剩下的就是单表的操作，他们存储了各自的用户相应的信息，t\_attach存储了文件上传的时候的文件名，t\_log存储了用户的操作记录日志，

他们之间的详细关系见图5.3.1

(文中图样式：图中文字五号宋体；图名在图的正下方，五号黑体)

6 系统实现